Walkie-Markie: Indoor Pathway Mapping Made Easy

Guobin Shen,[†] Zhuo Chen,[‡] Peichao Zhang,[‡] Thomas Moscibroda,[†] Yongguang Zhang[†]
Microsoft Research Asia, Beijing, China
[†]{jackysh, moscitho, ygz}@microsoft.com, [‡]{czxxdd, starforever00}@gmail.com

Abstract

We present Walkie-Markie – an indoor pathway mapping system that can automatically reconstruct internal pathway maps of buildings without any a-priori knowledge about the building, such as the floor plan or access point locations. Central to Walkie-Markie is a novel exploitation of the WiFi infrastructure to define landmarks (WiFi-Marks) to fuse crowdsourced user trajectories obtained from inertial sensors on users' mobile phones. WiFi-Marks are special pathway locations at which the trend of the received WiFi signal strength changes from increasing to decreasing when moving along the pathway. By embedding these WiFi-Marks in a 2D plane using a newly devised algorithm and connecting them with calibrated user trajectories, Walkie-Markie is able to infer pathway maps with high accuracy. Our experiments demonstrate that Walkie-Markie is able to reconstruct a high-quality pathway map for a real office-building floor after only 5-6 rounds of walks, with accuracy gradually improving as more user data becomes available. The maximum discrepancy between the inferred pathway map and the real one is within 3m and 2.8m for the anchor nodes and path segments, respectively.

1 Introduction

Accurate and inexpensive indoor localization is one of the holy grails of mobile computing, as it is the key to enabling indoor location-based services. Despite very significant research effort, relatively little has actually been deployed at scale. One reason is that a common and critical assumption of existing approaches – the availability of a suitable localization map – is hard to fulfill in practice. For instance, WiFi triangulation or fingerprinting based approaches for indoor localization rely on a priori AP position information, or a signal strength map to function properly [4, 13, 24]. Such maps are typically constructed via dedicated, often labor-intensive, data-gathering processes that map radio signals onto an indoor map that *geographically reflects the physical layout* of the building. Several recent efforts aimed at alleviating the pain of radio map construction require knowledge of the real floor plans [26, 38]. Similarly, tracking based localization also requires accurate indoor maps (e.g., floor plans or pathway maps) to constrain the drifting of inertia sensors [36, 37]. Such indoor maps are difficult to obtain in general, as they may belong to different owners, may be outdated, and many legacy buildings simply do not have them at all.

In this paper, we try to fundamentally rethink the assumption and ask the question: can we build an indoor map *without any prior knowledge* about the building? In particular, we are interested in building *pathway* maps because they provide a natural framework for localizing users and points of interest (POIs) as people usually move along pathways and indoor POIs are connected via pathways. A pathway map can also serve as the basis for other maps specific to other localization approaches, or can be used as a building block to construct semantically richer maps for users, for example through automatic location detection (e.g., [3]) or crowdsourced user annotation. Finally, we seek a technology that allows to obtain such pathway maps *at scale*, say for millions of buildings across the world, including shopping malls and office buildings.

We address these problems in *Walkie-Markie*, a system that automatically generates indoor pathway maps from traces contributed by mobile phone users. The system uses crowdsourcing to generate the pathway map of unknown buildings without requiring any a-priori information such as floor-plans, any initial measurements or inspection, and any instrumentation of the building with specific hardware. The only assumption Walkie-Markie requires is that there exists a WiFi infrastructure in the building that is to be mapped. AP locations do not need to be known; instead APs must merely exist for the system to work.

Walkie-Markie is based on two key observations. First, a modern mobile phone can dead reckon the user's movement trajectory from its inertial measurement units (i.e., IMU sensors, including accelerometer, magnetometer and gyroscope) [21,26,28,36]. The idea is that if sufficiently many users walk inside a building and report their trajectories, we can infer the pathway map. The challenge is that IMU-based tracking is accurate only initially as it suffers from severe drift: rapid error accumulation over time. Moreover, to generate maps at scale via crowdsourcing, we must deal with trajectories from different users, who may start their walks from anywhere, at different stride lengths, varying speed, etc. Second, WiFi networks have been widely deployed, from office buildings to shopping malls. WiFi has been successfully used by fingerprinting-based localization schemes, and combined WiFi and IMU-tracking solutions have also been proposed, e.g., [5,11,26,31,38]. However, there are well-known practical concerns when using WiFi for localization: signals fluctuate significantly during different times of day, different phones can have different receiver gains (i.e., device diversity) [14,34], and readings also vary depending on how people place their phones e.g., in hand, in pocket, or in backpack (i.e., usage diversity) [19].

Walkie-Markie consists of mobile clients on users' mobile phones and a backend service in the cloud. When participants walk, the client collects the rough trajectory information (step count, step frequency, and walking direction) as well as periodic WiFi scan results. The backend service fuses these possibly partial user traces (w.r.t the overall internal pathways) and generates the pathway map.

Central to Walkie-Markie is the *WiFi-defined landmark* (*WiFi-Mark*), which is a novel way to exploit the widely-deployed WiFi infrastructure to establish accurate and stable landmarks, which serve to anchor the various partial trajectories. A WiFi-Mark is defined as a pathway location at which the *trend* of received signal strength (RSS) from a certain AP reverses, i.e., changes from increasing to decreasing, as the user moves along the pathway. We show in this paper that such WiFi-Marks based on the RSS *trend* (instead of *the face RSS value* used in previous works) overcomes the aforementioned challenges in leveraging WiFi signals and yields highly stable and easily identifiable landmarks. WiFi-Marks are determined by the relative physical layout of the AP and the pathway, and are thus location invariant. Moreover, a single AP often leads to multiple uniquely identifiable WiFi-Marks, leading to a higher density of WiFi-Marks.

WiFi-Marks allow us to overcome two key problems in mapping buildings: i) merging the large volumes of crowdsourced (partial) trajectories and ii) bounding the tracking error and drift of IMU sensors. Being location-invariant, WiFi-Marks yield the common reference points for fusing snippets of user trajectories. With more user trajectories, the noise tend to cancel each out, which leads to more accurate displacement measurement between WiFi-Marks. Thus, mapping accuracy gradually improves as more data becomes available. IMU-based tracking suffers notoriously from rapid error accumulation as distance increases. WiFi-Marks also help with the drift problem of IMU-based tracking by bounding the distances between which IMU-based tracking must be relied upon.

Another ingredient of Walkie-Markie is a novel graph embedding algorithm, *Arturia*, that fixes WiFi-Marks to "known" 2D locations respecting the constraints suggested by the user trajectories. The resulting pathway map naturally reflects the physical layout. Arturia differs from existing embedding algorithms in that it uses measured *displacement vectors* as opposed to distances between nodes as input constraints. After WiFi-Marks are properly placed on the 2D plane, the pathway map is generated by connecting the embedded WiFi-Marks with corresponding user trajectories. The obtained pathway maps can be used by users to localize themselves by adding the displacement to the position of the last encountered WiFi-Mark. The pathway maps can also be used to generate other localization maps such as radio maps.

We have implemented Walkie-Markie and evaluated it in an office building and a shopping mall. Our experimental results show that we can achieve mapping accuracy within 3 meters by merging enough user trajectories (each as short as one minute) equaling to 5-6 rounds of walking. The mapping accuracy gradually improves and stabilizes after about 1-2 times more walking time along the same paths. Additional experiments on localization using pathway as well as radio maps produced by Walkie-Markie show that the average and 90 percentile localization errors are 1.65m and 2.9m, respectively, when using displacement from the last WiFi-Mark using the pathway map.

2 Problem and Challenges

Problem Statement: Indoor localization results are meaningful only when associated with corresponding indoor maps (e.g., pathway maps) that geographically reflect the physical layout. However, in this context the availability of such maps has largely been taken for granted, often via assumptions. For instance, IMU-based tracking and localization systems have assumed accurate indoor maps (e.g., floor plans) to constrain drifting; WiFi-based localization systems *further* assume a-priori knowledge about AP positions or a radio signal map [4,13,24]. While there are many existing works trying to

reduce the dependency on such a-priori information (AP locations [6], radio signal map [12, 16, 22, 26, 38]), they all assume a known internal map of the floor, which are often not readily available.

In this paper, we remove this assumption and build an indoor pathway mapping systems without assuming *any* prior knowledge of the building. Our goal is to find a solution that works with existing infrastructure, is applicable to commercial mobile phones, and is "crowdsourceable" so as to scale to a large number of buildings. The only assumption we make is the *mere* existence of a WiFi infrastructure in the buildings to be mapped.

Challenges: Previous work has tried to combine WiFi signals and IMU sensing data [5,10,11,31]. The problem with IMU-based technologies is that they can track a user's trajectory at some accuracy for only a short period of time, and will drift severely as the walking time increases. This makes it hard to align multiple trajectories, and trajectories obtained from different users (with different start points) are even harder to combine into a whole pathway map. Leveraging WiFi also poses well-known challenges. Even though WiFi fingerprints are statistically locality-preserving [16, 24], an AP's coverage area is overly large for the desired accuracy of a useful internal pathway map. Typically, multiple pathways are covered by a single AP. The AP's position is also unknown. Furthermore, other challenges common to WiFi-based localization systems are: i) WiFi signal fluctuations due to ambient interference, multipath effect, and environmental dynamics such as the time-of-day effect; ii) device diversity with different receiver gains at different phones; and iii) device usage diversity caused by people placing their phones differently such as in hand, in pocket, in purse, or in a backpack. We note that usage diversity is rarely mentioned in the literature, but is a real impairing factor.

3 WiFi-defined Landmark

In real life, *landmarks* are often used to give directions. No matter how one detours, once a landmark is encountered, previous errors are reset. Using the same idea, we can leverage landmarks to constrain the drifting in IMU tracking, and to align different user trajectories. However, the challenge is the find landmarks that are perceivable by mobile phones without human intervention. Since mobile phones can sense the WiFi environment in the background, we would ideally like to identify landmarks based on WiFi signal. In this section, we show that–using the concept of *WiFi-Marks*–this is indeed possible in spite of the multitude of challenges mentioned above.

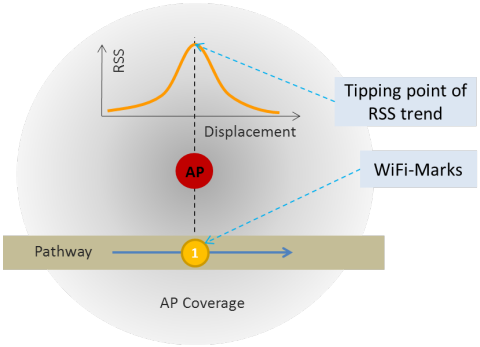


Figure 1: Illustration of WiFi-Marks, as determined by the relative physical layout of the AP and the pathways.

3.1 WiFi-Marks: Concept

Previous work on WiFi-based localization has used the received WiFi signal strength (RSS) directly. It turns out that this is the root cause of the aforementioned problems. The key insight is that significantly more stable landmarks can be obtained from an existing infrastructure by using WiFi signal strength *indirectly*: instead of looking at the face RSS values, we look at the *trend* of RSS changes.

Figure 1 illustrates the basic idea. A user is walking from left to right along a pathway covered by an AP. Initially we see RSS increase as the user moves closer towards the AP. When the user walks past the point from which the distance to AP increases, the RSS trend reverses. In theory, this *RSS trend tipping point* (RTTP) should correspond to a fixed position on the pathway that is closest to the AP in terms of signal propagation.

The key appeal of examining the RSS trend instead of taking individual RSS readings is that it may solve the device and usage diversity problems: no matter what make and model of the phone, what time of the day, and how the phone is kept with respect to its user, the RTTP should occur at around the same location. Through detailed experiments, we argue in Section 3.3 that locations where the RSS trend of a certain AP changes are excellent candidates for landmarks. We call these points WiFi-defined landmark, or short WiFi-Marks (WM) hereafter.

3.2 WiFi-Marks: Identification

As a landmark, each WiFi-Mark should be uniquely identifiable. Depending on how the coverage area of an AP intersects with the pathways, it is possible and in fact quite likely that one AP will generate multiple WiFi-Marks (see Figure 2). Hence while BSSID (the MAC address of the AP) can uniquely identify the master AP, it alone is insufficient to uniquely identify a WiFi-Mark since there can be multiple pathways under the cover-

age of the same AP. Therefore, we need to use additional information to differentiate different WiFi-Marks of the same master AP.

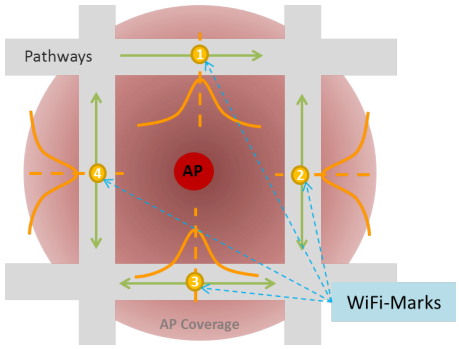


Figure 2: Possibly multiple WiFi-Marks for the same AP.

In Walkie-Markie, we identify a WiFi-Mark by the following three-tuple:

$$WM \triangleq \{BSSID, (D_1, D_2), \mathcal{N}\}$$

where *BSSID* is the ID of the master AP, D_1 and D_2 are the steady walking directions approaching and leaving the RTTP, respectively. They can be obtained from the phone's magnetometer. $\mathcal{N}$ is the set of neighboring AP-s' information, including their BSSID and the respective RSS differences to that of the master AP.

The walking direction information (D_1, D_2) is adopted to differentiate pathways and turns. For example, Figure 3 shows the possible RTTPs for AP_1, under different walking patterns. With directions, we can readily differentiate RTTP 1, {2,3}, 4, and 5. In addition, the direction can be used to disqualify some erroneous RTTP detections when the user makes a U-turn (e.g., RTTP 6 and 7). Not identifying such "U-turn RTTPs", could add significant noise to the system.

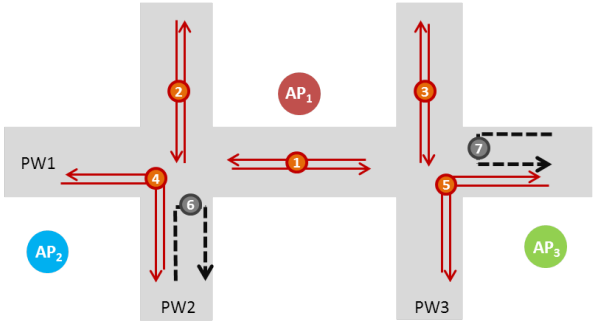


Figure 3: Multiple RTTP possibilities for AP1 under different walking patterns illustrated by arrows.

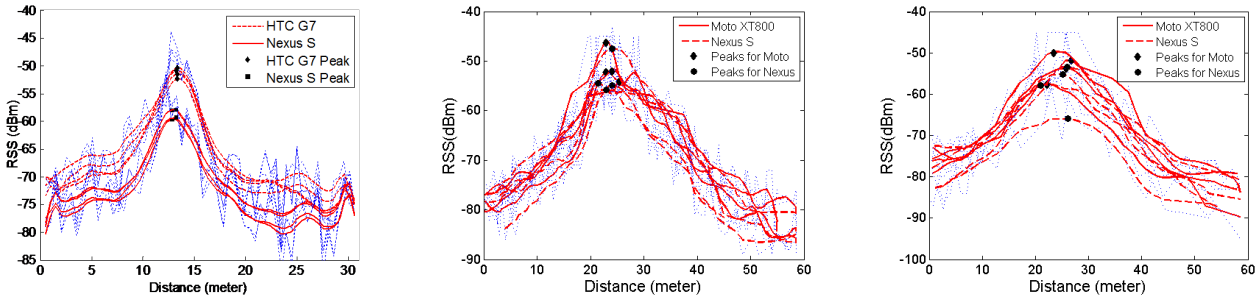
RTTPs with similar (D_1, D_2) can arise from parallel corridors (e.g., RTTP 2 and 3 in Figure 3) or similar turning styles. To further differentiate such RTTPs, we leverage neighborhood AP information. In the same example, RTTP 2 may see AP2 only and RTTP 3 sees AP3 only. Even if they see the exact same set of APs, there is still a good chance that the relative RSS values will be different due to the difference in distance to each AP. Note that it is important to use the RSS *differences* to the master AP's RSS instead of their real RSS values to avoid the device diversity problem. From the radio propagation model [1], it can be verified that RSS differences between multiple APs are not affected by the receiver gain for a device.

Due to sensing noise, D_1, D_2, and $\mathcal{N}$ of a given WiFi-Mark can be slightly different each time the WiFi-Mark is measured. Therefore, we employ a *WiFi-Mark clustering* process (see Section 6). There are further unreliable RTTP detections, such as when a user is not walking straight or steadily (e.g., zigzagging) or when the phone's position changes rapidly (e.g. taken out of the pocket). Our system therefore accepts an RTTP as a WiFi-Mark only if the IMU sensor indicates a stable walking motion and no U-turn is detected during the measurement process.

3.3 WiFi-Marks: Stability

Evaluation Scenarios: The indoor radio environment is complex and often deviates significantly from ideal propagation models. To verify the stability of the RTTPs in practice, we conduct the experiments using different devices (HTC G7, Moto XT800, and Nexus S), at different time of day (morning, afternoon, evening, and midnight), and with the phones held at different body positions (hand, trouser pocket, purse, and backpack). All of these are important factors affecting RSS. In addition, we perform experiments in two buildings to demonstrate the generality of our approach.

We present two sets of experiments. In the first set, we walk and wait, i.e., wait to ensure a complete scan of all WiFi channels before walking to a next collection point. This represents an ideal case. In the second set, we walk continuously at slow or normal speed without waiting for WiFi scans to complete. Figure 4 shows the curves of collected RSS values and the locations of the detected WiFi-Marks. From the figure, we can see that the RSS values from different devices are evidently different, and the same is true for the same device at different time of day, or at different body positions. In contrast, the increasing and decreasing RSS trends are always easily identifiable, and the WiFi-Mark positions are not only highly clustered and stable, but also consistent between the two devices. Taking the normal walking case as an example, the average position deviations are 1.3m and 2.9m for Moto XT800 and Nexus S, re-



(a) Walk and wait. (b) Slow walking. (c) Normal walking.

Figure 4: RSS curves for one AP along a corridor using two phones. Blue dotted lines and red solid lines are the raw and filtered RSS curves (see Section 6). Multiple same type of lines are measurement from different time of day. In (a), all phones were held in hand. In (b) and (c), Moto was held in hand and Nexus S in trouser pocket.

spectively, while the mean center position offset is only 2.7m between devices. In the ideal case, the deviations are even better. The reason is that because of the relatively long WiFi scanning time in today's mobile phone (usually about 1.5s), the user may have already walked a few steps during a scan.

Stability Evaluation: We also conduct controlled experiments in larger areas with more pathways, still using various devices and walking at various speeds and at different times of day. For each different setting we collect data over 5 rounds and calculate the statistical deviation in WiFi-Mark position. We note that the peak RSS value at RTTPs are not all strong, some being as weak as -75 dBm.

Figure 5-(a) shows the cumulative distribution function (CDF) of the deviations for the different settings. We can see that for over 90% of WiFi-Marks, the deviations are within 2.5m, and about 70% are within 1.5m in all cases. We further study whether WiFi-Marks detected with different settings are consistent, using the center offset of WiFi-Mark clusters. Figure 5-(b) shows the CDF of the center offsets. They are indeed consistent: over 95% of the offsets are within 2.5m and over 75% are within 1.5m. These results demonstrate that WiFi-Marks are stable and robust across various dimensions, and thus have ideal properties to be landmarks for our indoor pathway mapping purpose.

4 Walkie-Markie: Overview

With WiFi-Marks, we now have the common reference points for fusing crowdsourced user trajectories together. Walkie-Markie consists of a client–an application running on users' mobile phones–and the backend service running in the cloud. The overall architecture is shown in Figure 6.

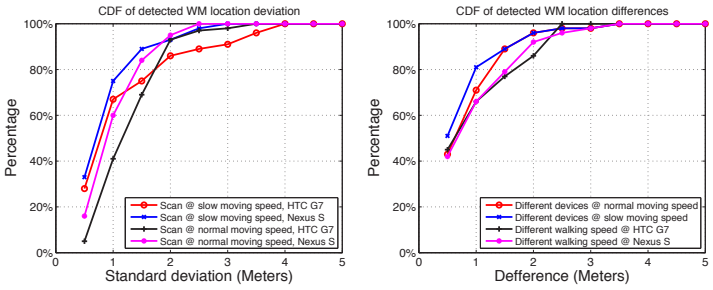
A Walkie-Markie client works as follows: a background motion state detection engine monitors users' motion states periodically. When the user is detected in *walking* state, IMU-based tracking is activated and the instantaneous walking frequency and direction of each step is recorded for displacement estimation. At the same time, WiFi signal scanning is performed opportunistically. If a WiFi signal is detected and the device has not associated with an AP, the WiFi-Mark detection process is activated. Information about the detected WiFi-Marks and estimated displacements between neighboring WiFi-Marks are stored, and later sent to the backend service.

The Walkie-Markie backend service listens to WiFi-Mark updates from all clients. Upon receiving WiFi-Mark updates, it examines if their master APs are new or already existing. Updates with new master APs are recorded and aged to mitigate the impact of transient APs (e.g., mobile APs). For existing ones that are old enough, their neighborhood consistency is further checked to ensure they are not relocated APs, which would be treated as new APs. Then a clustering process is executed to cluster different detections of the same actual WiFi-Marks. Each cluster is then assigned one coordinate by the Arturia engine. Finally, with WiFi-Marks positioned at the right places and user trajectories connecting them, the backend service can generate the desired pathway maps.

5 WiFi-Mark Positioning

WiFi-Marks (or landmarks in general) serve their purpose as a reference points only once we can place them at a known location. For this reason, we need to assign coordinates to WiFi-Marks, which is a classical node embedding problem in the network coordinate and localization literature.

Distance vs Displacement: Previous node embedding work has unanimously assumed scalar distances (e.g., via a direct distance measurement or the shortest path) between nodes [9, 23, 30]. However, in our case, users may



(a) CDF of WM position deviation (b) Consistency of WM positions

Figure 5: Statistics on WM positions.

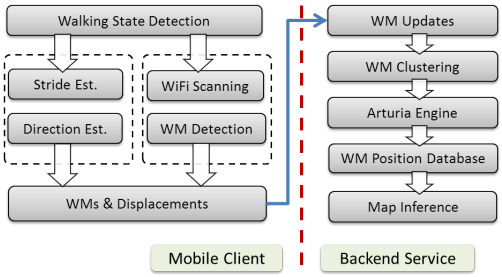


Figure 6: Walkie-Markie system architecture.

not always take the shortest path and, in fact, the internal floor layout may even prevent people from taking shortest paths (e.g., two nearby WiFi-Marks blocked by a wall or a locked door). If multiple paths exist, taking different paths will lead to different distances. These factors often lead to severe violations of the triangle inequality, which lies at the heart of existing embedding algorithms. Consequently, using distances between WiFi-Marks is insufficient and the *displacement vector* (i.e., both the distance and the direction, obtainable from IMU sensors) between WiFi-Marks has to be used.

Using direction information in addition to distance is fundamental, because it can largely avoid the "fold-freedom" problem of the embedding process [25], and dismiss flip and rotational ambiguities. The *only* remaining translational ambiguity can be fixed by fixing any anchor point with an absolute location (e.g., entrances or window positions of a building with GPS readings). In addition, using direction information also requires fewer measurements: only N unique displacement measurements are required to localize N WiFi-Marks, whereas $3N-6$ unique measurements would be required when using distances only (in which case the results would still suffer from flip and rotational ambiguities). Thus, using displacement vectors enables faster bootstrapping and is highly desired for a crowdsourcing system.

5.1 Arturia Positioning Algorithm

In our system, a major challenge is the inaccuracy of IMU-based displacement measurements (e.g., errors in stride length and/or direction estimation). To compensate these errors, we design a new embedding algorithm, *Arturia*, that handles noisy IMU measurements and assigns optimal coordinates to WiFi-Marks. Arturia is based on the *spring relaxation* concept, where each edge of the graph is assumed as a spring and the whole graph forms a spring network.

Building the Graph: An edge (hence, a spring) is added between two specific WiFi-Mark nodes as long as there exists a real user trajectory in between. The rest length of the spring (i.e., the constraint) is the real displacement measurement from user trajectory. Multiple edges between a pair of nodes are possible if there exist multiple user trajectories. In this way, we ensure that more frequently encountered WiFi-Marks will have more accurate coordinates as compared with the alternative strategy that uses a single average edge.

Realizing the Graph: With the spring network, our goal is to minimize the overall residual potential energy E, which is a function of the discrepancy between the calculated distance (i.e., actual length of the spring) and the real measurement (i.e., rest length of the spring). Our solution is to adjust the node's position as if it were pushed or pulled by a net force from all connecting neighboring springs. Arturia works as follows:

Initialization: We may randomly assign all nodes's initial coordinates, or simply to the origin. But for updates due to new incoming data, the previous coordinates are used for faster convergence and better consistency, i.e., minimal adjustment to the previous graph.

Iteration: At each iteration, adjust the coordinates for each node according to the compound constraints of the neighboring nodes. Let $\hat{p}_i$ be the current coordinate of node i. We have $\vec{d}_{i,j} = \hat{p}_i - \hat{p}_j$ as the current displacement vector between node i and a neighboring node j. Assume there are $N_{e,i,j}$ real measurement constraints between node i and j, and let $\vec{r}_{i,j,k}$ be the k^{th} constraint. Then the adjustment vector is calculated as

$$\vec{\varepsilon}_{i,j} = \sum_{k=1}^{N_{e,i,j}} (\vec{r}_{i,j,k} - \vec{d}_{i,j}) \qquad (1)$$

The gross adjustment vector $\vec{F}_i$ is obtained by summing up $\vec{\varepsilon}_{i,j}$ over all neighboring nodes, i.e., $\vec{F}_i = \sum_j \vec{\varepsilon}_{i,j}$. Then, node i's coordinate is updated as $\hat{p}_i = \hat{p}_i + \vec{F}_i$.

The step size of the adjustment (i.e., $|\vec{F}_i|$) plays a critical role in the convergence speed: large adjustment steps may lead to oscillation while small adjustments will converge slowly, as also observed in [9]. To obtain a suitable step size, we empirically amortize the adjustment vector

according to $N_{e,i}$, the total number of edges to all neighbors of node i. That is, $\vec{F}_i = \frac{1}{N_{e,i}} \sum_j \vec{\varepsilon}_{i,j}$.

Termination: For each node i, the local residual potential energy E_i is calculated as $E_i = \sum_j |\vec{\varepsilon}_{i,j}|^2$. System residual potential energy is then $E = \sum_i E_i$. This value tends to increase with additional edges of the spring network. To obtain a universally applicable termination criterion, we use the normalized potential energy $\bar{E} = E/N_e$ with N_e being the total number of constraining edges. The iteration will terminate when the change of $\bar{E}$ falls below a small pre-determined threshold.

Algorithm Comparison: The spring relaxation concept has previously been adopted, e.g. in [9, 15, 25]. The major difference is that the local adjustment (i.e., $\vec{F}_i$) in each iteration has *direction* information and will always move closer to the target coordinates in Arturia. This is not the case in other algorithms where the moving direction is calculated based on the noisy, intermediate coordinates. Figure 7 illustrates this difference between Arturia and the Vivaldi [9] algorithm for an intermediate adjustment step to Node 3. We can see that in Arturia, the net force of the adjustment points directly to Node 3's target position, while in Vivaldi it does not. The reason is that the constraints in Arturia are *displacement vectors* (e.g., $\vec{r}_{3,1}$ and $\vec{r}_{3,2}$) with direction information, while in Vivaldi they are scalar distances (e.g., $|\vec{r}_{3,1}|$ and $|\vec{r}_{3,2}|$).

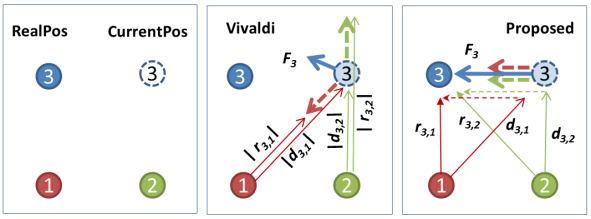


Figure 7: Illustration of an intermediate adjustment step of Vivaldi [9] and Arturia.

5.2 Arturia Evaluation

We evaluate Arturia with simulations. We randomly deploy N nodes in a 100×100 square area. For each node, we build n edges to n random neighboring nodes. For each edge, the direction is adjusted by a random number within ± 30 degrees, while the distance (i.e., the magnitude of displacement) is randomly adjusted by within ± 10 percent. These numbers reflect the real displacement estimation error ranges.

Anti-folding Capability: As mentioned, using direction helps to avoid "fold-freedom" issues. We demonstrate this by comparing the snapshots of intermediate steps of Arturia against those of Vivaldi and AFL (see Figure 8). We see that after 100 iterations, the nodes are still heavily folded in Vivaldi. AFL is better than Vivaldi in shape, but at a wrong scale. For Arturia, the nodes are almost in correct positions after only 30 iterations.

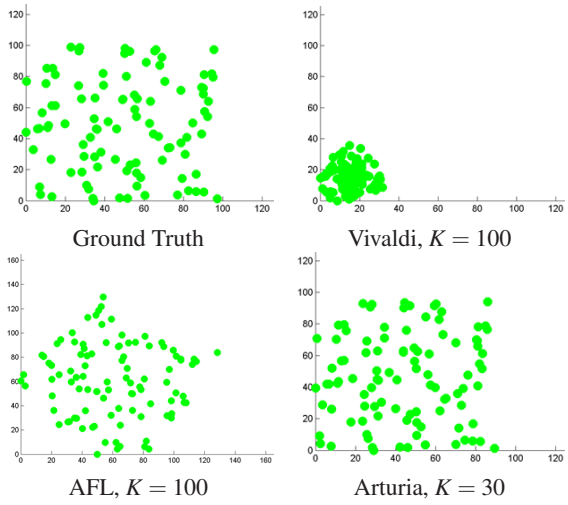


Figure 8: Snapshots of node positions at the different iterations for Vivaldi, AFL and Arturia.

Speed and Accuracy: We study the convergence speed and the resulting accuracy of different algorithms by varying the parameters N and n. Each experiment is repeated 10 times and average results are reported. Note that in the simulation, we have used the magnitude of displacement as the distance for Vivaldi and AFL to ensure the obeyance of triangular inequality, i.e., all nodes are mostly localizable.

The speed is measured as the number of iterations. For the accuracy metric, we adopt the Global Energy Ratio (GER) because it captures the global structural property [25]. GER is defined as the root-mean-square normalized error value of the node-to-node distances, i.e., $GER = \sqrt{\Sigma_{i,j:i<j} \hat{\varepsilon}_{ij}^2 / (N(N-1)/2)}$ where N is the total node number and $\hat{\varepsilon}_{ij} = |\Delta\vec{d}_{ij}|/|\vec{d}_{ij}|$ is the normalized node distance error.

Table 1 shows the results. We see that the proposed Arturia algorithm is significantly better than the other two algorithms in terms of both convergence speed and accuracy. In general, with higher connectivity, both speed and accuracy improve for all three algorithms. This is due to larger damping effects resulting from more densely interconnected springs. However, even with dense connectivity, the accuracy of Vivaldi is poor because of heavy folding. AFL works better by finding better initial positions. In our target scenario, the node connectivity cannot go very high since there will rarely be direct displacement measurements between faraway WiFi-Marks. This highlights the advantage of Arturia in

N	n	Speed (Iterations)			Accuracy (GER)		
		Viv.	AFL	Art.	Viv.	AFL	Art.
100	4	319k	193k	763	.687	.241	.0091
	6	38k	27k	450	.660	.106	.0072
	8	11k	2244	232	.615	.015	.0061
	10	6954	971	170	.614	.012	.0056
200	4	340k	334k	1552	.745	.279	.0068
	6	42k	19k	706	.736	.060	.0053
	8	20k	3299	441	.710	.012	.0049
	10	10k	1552	339	.699	.010	.0046

Table 1: Speed and Accuracy comparison of Vivaldi, AFL, and Arturia. N is the node number and n is the node connectivity degree.

the context of Walkie-Markie.

6 System Implementation

We have implemented the Walkie-Markie system, with mobile client on Android phones and backend services as Web Services. In this section, we detail a few key components.

WiFi-Mark Detection: In mobile client, the collected RSS value is first smoothed over a 9-point weight window in a running fashion to detect WiFi-Marks. The weight window is empirically set as a triangle function ($\{0.2, 0.4, 0.6, 0.8, 1, 0.8, 0.6, 0.4, 0.2\}$). We tested other window functions (e.g., cosine, raised cosine) and found not much difference in detection accuracy. Then, the trend detection is done by taking derivatives of the smoothed RSS curves, i.e., the differences between neighboring points. The consecutive positive and negative spans of the differences are identified and the corresponding walking directions are checked. If there are no U-turns and the trend change is significant as controlled via a threshold (e.g., 5 dBm), the point with the peak (filtered) RSS during the trend transition is selected as a WiFi-Mark.

Displacement Estimation: Displacement between WiFi-Marks is estimated from user trajectories by accumulating the displacement of each step. Step displacement carries stride length and walking direction and is captured by IMU sensors. Many techniques exist for stride length estimation [17, 29, 32]. We chose a simple frequency-based model by Cho *et al* [7]: $stride\_len = a \cdot f + b$ with f being the instantaneous step frequency, and a, b being parameters that can be trained offline. However, model parameters are specific to a user's walking conditions, e.g., parameters trained from wearing sport shoes will not work well when wearing high heels. Improper parameters will lead to large estimation error.

Interestingly, leveraging common WiFi-Marks among user trajectories, we can avoid the error-prone stride length estimation and instead rely on simpler and more robust *step counting* under regular walking, which can be easily be done from the regularity of the IMU data. We first randomly select a user and treat her stride length as the benchmark unit (BU). We then normalize other users' stride against hers using partial trajectories between common WiFi-Marks and obtain a normalization factor θ. This normalization process is transitive. Ultimately all users will normalize their traces to the same BU and obtain their respective θs. Then we obtain the average normalization factor $\bar{\theta}$. The product of $\bar{\theta}$ and the BU will be the real stride length of the "average" user, to which we can assign the demographic average stride length.

Walking Direction Estimation: We use the magnetometer and the gyroscope to obtain the walking direction and the turning angles, similar to [18, 21]. Unlike step detection and stride length that is determined on a per-step basis, the direction of each step needs to be determined by considering those of neighboring steps because magnetometer readings are sometimes not stable due to disturbance of local building construction or appliances, and the gyroscope may drift over time. In our implementation, we simply discard portions of magnetometer data where drastic changes occur, and rely on the gyroscope to decide whether there is a direction change in that period. For the portions with stable magnetometer readings, we use a Kalman Filter to combine the magnetometer and the gyroscope readings to tell the user's walking directions.

WiFi-Mark Clustering: The backend service receives many crowdsourced trajectories and WiFi-Mark reports. Due to sensing noise and user motion, the same actual WiFi-Mark may be reported slightly differently in directions (D_1, D_2) and neighborhood ($\mathcal{N}$). We design a clustering process to detect the same actual WiFi-Mark: we first classify reported WMs with the same BSSID using D_1 and D_2. To accommodate sensing noise, the directions are considered the same if they are within ± 20 degrees. For those WMs with same BSSID and similar directions, a bottom-up hierarchical clustering process as in [6, 24] is applied on the neighborhood set $\mathcal{N}$. Initially, each WM is one cluster. Then the closest clusters are iteratively merged if their inter-cluster distance is smaller than a pre-determined threshold, which is set to 15 dBm as recommended in [24].

The inter-cluster distance is the average distance between all inter-cluster pairs of WMs. The distance between two WMs is defined over the RSS of the sensed APs ($\mathcal{A}$) as follows:

$$D_N(\vec{\mathcal{A}_i}, \vec{\mathcal{A}_j}) = \sqrt{\sum_{n=1}^{K} (a_n^i - a_n^j)^2 / K}$$

where $\vec{\mathcal{A}}_i$ are the RSS *differences* (to ensure device indifference) between the master AP and neighboring APs at WM_i, and K is the total number of unique APs detected at the two WMs. For orphan APs appearing in one WM's neighborhood but not the other, the RSS difference is set to peak RSS of the master AP minus -100 dBm. Finally, each WiFi-Mark cluster is treated as one node in Arturia and assigned one coordinate. All WiFi-Marks in the same cluster have the same coordinate.

Pathway Map Generation: With WiFi-Marks and connecting user trajectories, we design the following expansion and shrinking procedure to generate the pathway map systematically. Initially, user trajectories are partitioned into snippets delimited by WiFi-Marks. Snippets with U-turns are filtered out. The remaining trajectory snippets are calibrated by proportionally adjusting the length and direction of each step using the WiFi-Marks' coordinates assigned by Arturia (affine transformation). For each calibrated trajectory snippet, we first draw it on a canvas and further expand it to a certain width (i.e., from line to shape). Pixels being occupied are weighted differently according to their distances to the center line: the closer the pixels, the higher the weight. Due to the multiplicity of user trajectories, there may exist multiple snippets connecting the same two WiFi-Marks. Thus, expanded snippets will overlay together and the weight of overlapping pixels are summed up. The expansion process will result in a fat pathway map. A shrinking process is then applied to prune away those outer pixels whose weights are less than a threshold. As some WiFi-Marks may be encountered more frequently than others, we adapt the threshold as a percentage to the maximum weight in the local neighborhood. Finally, we remove isolated pixels and also smooth the edges of the resulting shrunk pathway map.

Note that the pathway map generated from above expansion-shrinking process is a bitmap. It is also possible to generate a vector pathway map as all the turns can be effectively determined from user trajectories. We adopt bitmap in this paper for its immediacy in visually reflecting the quality of users' trajectories.

Practical Considerations: In our implementation, we have considered other important issues to build a practical crowdsourcing system.

Robustness: We have designed two mechanisms to improve the robustness of the system. First, the backend service implements an enrollment selection mechanism. WiFi-Marks from new master APs are recorded and will be incorporated into the Arturia positioning engine only when the AP becomes sufficiently aged. This is to counter transient WiFi-Marks, e.g., those caused by mobile APs or WiFi hotspot created on mobile phone through tethering. Relocated APs are detected via neighborhood (carried in WiFi-Mark reports) consistency and treated as new APs. Second, to mitigate the impact of outlying WiFi-Marks, e.g., resulting from transient mobile AP or wrongly detected due to magnetometer malfunction, we enroll a WiFi-Mark cluster only when it has a sufficient number of members (e.g., three).

Energy consumption: IMU-sensing consumes little energy, especially at low sampling rate (e.g., 10Hz in our case). Our preliminary test shows that 10Hz IMU sensing shortens the depletion time of a fully charged battery from 18.3 hours to 17.8 hours. We reduce data communication to the server by performing step detection and WiFi-Mark detection entirely on the mobile phone. The final communication data rate is about 1KB every 100 steps. Note that it can be delayed and piggybacked on other network sessions. The major energy consumption is from WiFi scanning. To work around, our client triggers WiFi scanning only when the user is walking (detected from low duty cycled IMU sensors), and we task a user to collect just a few minutes of walking data. As shown in Section 7, even short trajectories can still be used to infer pathway maps.

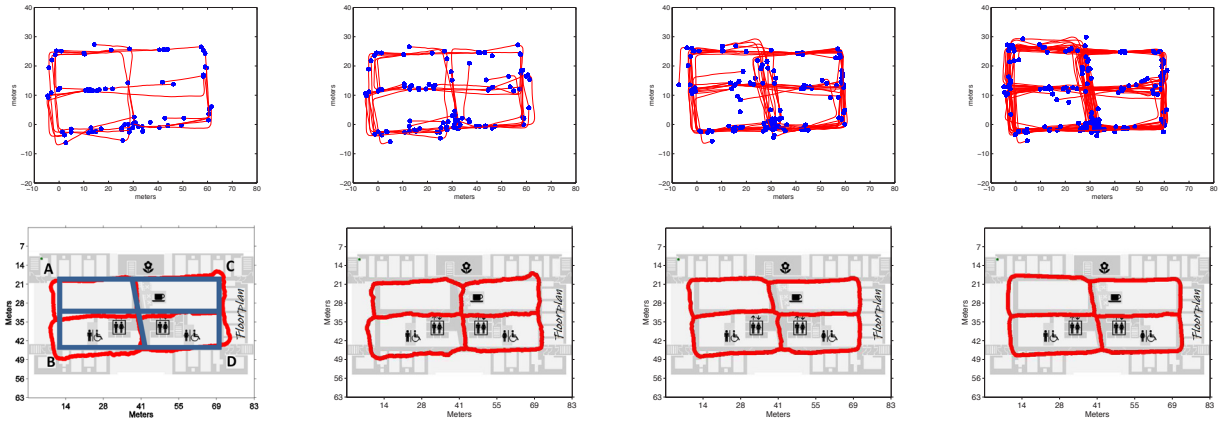
7 Walkie-Markie System Evaluation

7.1 Visual Comparison

Before presenting quantitative evaluation results, we first visually examine the inferred pathway map with the ground truth or reference floor plan. This will give us a general feel for Walkie-Markie's practicality.

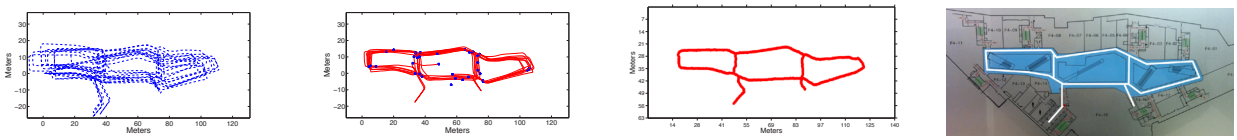
An Office Floor: We first show the study in our office floor for which we have the groundtruth floor plan. The internal layout consists of meeting rooms, offices, cubicle areas, and relatively large open areas in the middle. The experiment floor size is 3,600m^2 and the total internal pathway length is 260m. Figure 9 shows the aligned user trajectories and the inferred pathway maps under different amounts of user trajectory data. The stars in user trajectories are the detected WiFi-Marks. As expected, the quality of the resulting pathway map improves with more user data. After 50 minutes of random walk, the resulting map is already very close to the real map shown in the bottom-left figure.

A Shopping Mall Floor: We also study a nearby shopping mall. There is no managed WiFi LANs, but many isolated WiFi islands deployed by coffee shops or from POS machines. The floor has an irregular layout and the internal pathway length is roughly 310m. We walked about 10 rounds for about 40 minutes with a Nexus S phone. The results are shown in Figure 10. The first two figures show the raw IMU-tracked user trajectories



(a) after 20min walk. (b) after 30min walk. (c) after 50min walk. (d) after 100min walk.

Figure 9: Aligned user trajectories and generated pathway maps at different amount of user trajectories.



(a) normal IMU-tracking (b) with WM alignment (c) Inferred pathway map (d) Picture from flyer.

Figure 10: The picture and generated pathway map for a real shopping mall.

and those aligned with WiFi-Marks. The third figure shows the inferred pathway map. Unable to obtain a groundtruth floor plan, we took a picture of an emergency guidance map and highlighted the pathways in the last figure. We see that the pathway map generated by Walkie-Markie is visually very close to the real one.

7.2 Quantitative Evaluation

We conduct experiments in our office building, for which we have the groundtruth floor plan.

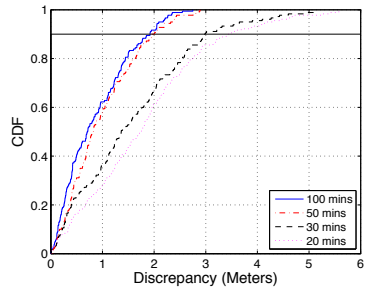
Data Collection: We have collected data from seven users, six male and one female, with heights range from 158cm to 182cm. A stride length model is trained for each user. We asked them to walk normally and cover all the path segments in each round, but they could start anywhere. Three phone models (Nexus S, HTC G7, and Moto XT800) were used. The phones were held in hand in front of body, hip-pocket, and also a backpack. In total, the users walked 30 rounds for about two hours.

In real crowdsourcing scenarios, users may walk only a portion of all pathways, or we may need to discard portions with irregular walking, or a user may only want to be tasked for a short time for consumption of energy consumption. To simulate these constraints and see if short trajectories are still useful, we chop the complete user trajectories into one-minute snippets, and randomly select a certain number of such snippets to infer the map. Results reported below are averaged over 10 such experiments.

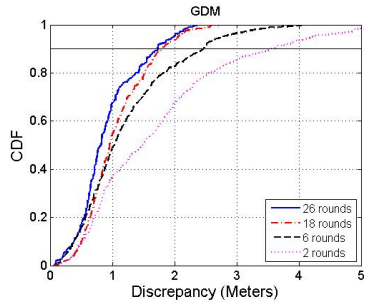
Performance Metrics: To quantify the quality of the inferred pathway map, we use the following metrics.

- *Graph Discrepancy Metric* (GDM): This metric reflects the differences in the relative positioning among anchor nodes, i.e. singular locations such as crosses or sharp turns. Like GER, we compare the Euclidean distances among all node pairs using coordinates from respective maps.
- *Shape Discrepancy Metric* (SDM): This metric quantifies differences between the shape of inferred paths and real ones. Path segments between corresponding anchor nodes are uniformly sampled to obtain a series of sample points. The metric is defined as the distance between corresponding sampling points. Note the inferred map needs to be registered to the real map first by aligning at some anchor nodes.

Mapping Accuracy: Figures 11-(a) and (b) show the cumulative distribution (CDF) of GDM from different amount of trajectory data. We can see that the geometric layout of all anchors are well preserved with only 2-hour walking data. The maximum difference in distances between corresponding node pairs is about 3 meters, and the 90 percentile difference is around 2 meters. We also observe that the performance improves as more data becomes available. In addition, an accurate pathway

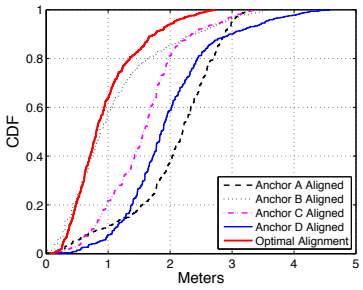


(a) Random snippets

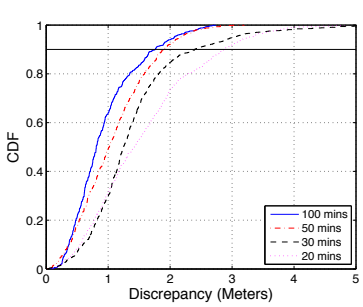


(b) Complete rounds

Figure 11: CDF of GDM.

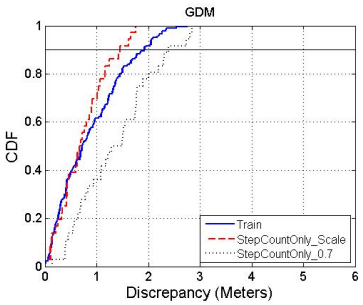


(a) Intuitive alignments

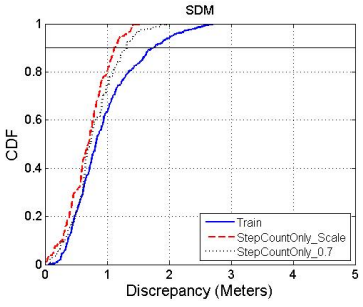


(b) Optimal alignment

Figure 12: CDF of SDM.



(a) GDM



(b) SDM

Figure 13: System performance using step count only.

map can be built from trajectories as short as one-minute walking, as long as we can obtain sufficiently many of them.

Comparing the curves with similar walking time (e.g., 100min vs 26 rounds) in the two subfigures, we can see that using complete trajectories leads to better performance. This is because chopping the walks into snippets reduces the displacement measurements between WiFi-Marks. In general, longer trajectories yields better performance.

In measuring SDM, we have different options to align the inferred map to the real map to fix the only remaining translational ambiguity. In reality, such alignment can be automatically performed by leveraging user trajectories that enter or leave the building. Here, we study the results by aligning at any outermost anchor point (e.g., Points A, B, C, D in the bottom-left figure in Figure 9), and also an optimal alignment at the geometric center of all anchors. In all experiments below, we have obtained 10 sample points on each path segment between two neighboring anchors.

Figure 12-(a) shows the CDF of SDM using 100 one-minute snippets. We can see that aligning at different points indeed leads to different performance. Nevertheless, the maximum difference among all the five alignment trials is small, within 1.3 meters. In the remainder of the evaluation, we use the optimal alignment. From Figure 12-(b), we see that the shape of inferred pathways agrees well with the shape of real ones. When over 50 minutes of walking data is used, the maximum path discrepancy is within 2.8 meters, and the 90 percentile error is within 1.8 meters.

Step Count Only: We stated above that Walkie-Markie can avoid error-prone stride length estimation. To verify this claim, we use only the direction and step count from the same set of user trajectories. Figure 13 shows the results. Since we do not know the demographic average step length, we scale the resulting shape to best fit the ground truth. This gives the upper bound of system performance. We also simply assign 0.7m as the demographic average step length and obtain the results. From the figure we can see that even using step count only leads to high accuracy maps. Comparing with the curve using the trained stride length model, we can see that the 90 percentile GDM is only slightly worse (within 0.4m) and the 90 percentile SDM is actually better by about 0.4m.

Impact of AP Density: Our office floor has a relatively dense AP deployment, about 21 APs covering an area of 3,600m^2. It is natural to conjecture that the performance of Walkie-Markie may be highly affected by the AP density. To study this impact, we emulate sparse deployments by randomly blanking out a certain percentage of APs, i.e., eliminating all the WiFi-Marks defined by those APs and their appearances in other WiFi-Mark's neighbor AP list.

Figure 14 shows the results with varying percentage

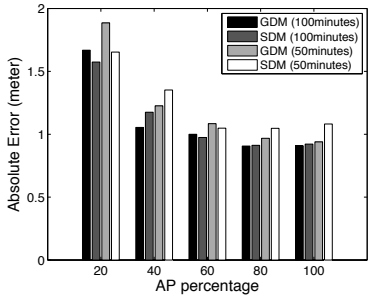


Figure 14: Impact of WiFi AP density.

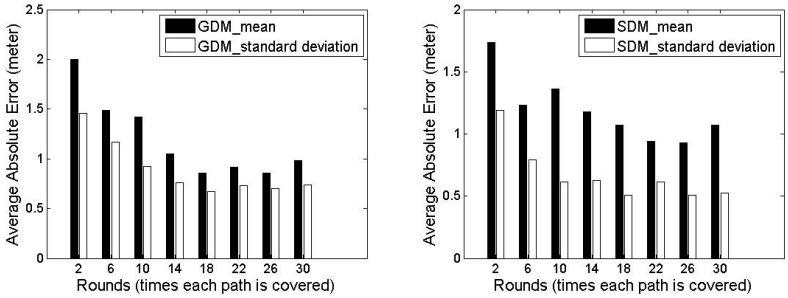


Figure 15: GDM and SDM statistics under different amount of trace data.

of remaining APs. In general, the performance degrades when the number of AP decreases. But for a dense deployment like our office building, the number of APs is more than enough for a good result. The result does not suffer if AP density is reduced to 40%. And even a further reduction to 20% degrade the mapping accuracy only slightly.

System Agility: We are also interested in learning how *agile* Walkie-Markie can construct a useful internal pathway map. System agility reflects the adaptation capability to the internal layout changes of a building. It is measured by the achievable GDM and SDM under different amount of user trajectories incorporated into the system. Figure 15 shows that both discrepancy metrics decrease with more data input, and the system converges quickly: with about 5 to 6 rounds of trajectories (i.e., visits per path segment), a highly accurate pathway map can already be inferred.

8 Application to Localization

Radio Map as Side Product: In Walkie-Markie, WiFi fingerprints are collected when the users walk. When the internal pathway map is generated, the position of each user step can be obtained from the calibrated walking trajectory. With reference to the timestamps of WiFi scans and steps, we can easily interpolate the position of each WiFi scan. As a result, we can generate a dense WiFi fingerprint map for free.

Localization: Both the resulting internal pathway map and the radio map can be used for localization purpose. For the former, we can localize a user by tracking the relative displacement since the last WiFi-Mark encountered, whose position is known. For the latter, we can apply any WiFi fingerprinting-based method such as the RADAR localization system [4]. For evaluation, we walked one round along the pathway in the office floor. During walking, we ensured every step to be at boundaries of carpet tiles. Thus fingerprints are collected at half-meter (i.e., the tile size) interval and their groundtruth positions are also known. We compare the localization results in Figure 16. We can see that Walkie-Markie outperforms RADAR, and more interestingly, the localization error is bounded. Quantitatively, the average and 90 percentile localization errors are 1.65m and 2.9m for Walkie-Markie, and 2.3m and 5.2m for RADAR. We note that the resulting accuracy is comparable with that reported in Zee [26], and slightly better than that from LiFS [38].

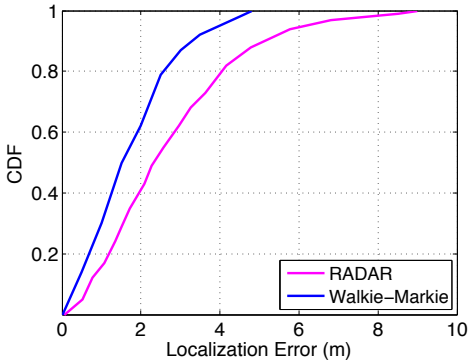


Figure 16: Localization results of Walkie-Markie and RADAR in an office floor, using crowd sourced map.

9 Discussion

Open Area: Our system works well for normal indoor pathways that are typically narrow (say a few meters), which helps ensuring regular user motion. For large open areas, the performance depends on how users walk. If most users walk along roughly the same path (e.g., from one entrance to another), Walkie-Markie will still work. In general, however, the performance may deteriorate as users may walk arbitrarily, which will cause noisy WiFi-Mark detection and clustering. For wide pathways, the inferred map tends to be thinner than the real ones. This is because we have assumed a point representation of a WiFi-Mark cluster, and we have also assumed the pathway to be around 2-meter wide pruning outer pixels in the shrinking process. We note that WiFi-Mark clusters

from wider pathway segments tend to be more diverged than those from thinner ones, we may leverage this fact to estimate the pathway width.

Multiple Floors: Users may walk across different floors using either elevators, escalators, and stairs. These motion states can be discriminated using accelerometer with advanced detection mechanisms [20,35], and can thus be excluded in the WiFi-Mark detection. Interestingly, these functional areas may serve as landmarks as they are stable and reliably detectable via phone sensors. Thus, they can also be incorporated into the Walkie-Markie system, and treated in the same way as WiFi-Marks by the Arturia engine. To discriminate different functional areas of the same type, we can use the covering WiFi APs.

Dedicated Walking *vs* Crowdsourcing: While Walkie-Markie is crowdsourcing-capable, it can also be used by dedicated or paid war-walkers. Dedicated walkers can walk longer and better traces, which leads to a higher efficiency in generating the desired maps (as shown in Figure 11).

10 Related Work

Although we focus on internal pathway mapping, Walkie-Markie is essentially a system of simultaneous localization and mapping (SLAM), which is heavily studied in the robotics field [33]. SLAM methods typically rely on visual landmarks or obstacles detected by camera, sonar or laser range-finders and on accurate kinematics of robots [2]. FootSLAM [28] uses shoe-mounted inertial sensors to construct the internal map. PlaceSLAM [27] further incorporates *manually* annotated places. In contrast, Walkie-Markie requires no special hardware and uses IMU sensors on commercial mobile phones, and requires no human intervention, which is necessary for a crowdsourcing system.

Escort [8] navigates users via the map built from other users' trajectories and instruments audio beacons to constrain IMU-tracking drift. Unloc [35] further explores various types of natural landmarks detectable from sensor readings, including the landmarks from WiFi networks. Their WiFi landmarks are determined as locations least similar (with ratio of common APs as the similarity metric) to all other places. Walkie-Markie does not need to instrument the environment, and uses the RSS trend to detect WiFi-Marks. This idea makes it robust to signal fluctuations, device diversity, and usage diversity, whereas how Unloc handles such practical issues was not reported. The detection is much simpler. In addition, unlike Unloc where multiple APs may determine one WiFi landmark, one AP may determine multiple WiFi-Marks in Walkie-Markie. Thus, we are able to find significantly more WiFi-Marks (e.g., over 100 WMs in one floor) than Unloc (e.g., around 10 WiFi landmarks and overall 40 landmarks in one building). One recent work [19] also exploits the point of maximum RSS, which bears similarity to WiFi-Mark. However, instead of exploiting it as a landmark, they use it to switch between two location inference modules. A dedicated training stage is required to obtain the locations of such maximum RSS points. Walkie-Markie builds the pathway map without pre-training.

There are several papers that combine WiFi and IMU-tracking for mapping purpose. WiSLAM [5] seeks to construct the WiFi radio map and uses the RSS values to differentiate different paths. WiFi-SLAM [11] uses a Gaussian process latent variable model to build WiFi signal strength maps and can achieve topographically-correct connectivity graphs. SmartSlam [31] employs inertial tracing, a WiFi observation model and Bayesian estimation method to construct the floor plan. LiFS [38] and Zee [26] seek to reduce efforts in generating the radio map, with the necessary aid of the actual floor plan. All these work has exploited the WiFi signal in the same way as other WiFi-based localization methods, and thus still face the same challenges, namely WiFi signal fluctuations, device diversity and usage diversity. Again, Walkie-Markie avoid such challenges by using RSS trend instead of face values.

11 Conclusion

We have presented the design and implementation of Walkie-Markie – a crowdsourcing-capable pathway mapping system that leverages ordinary pedestrians with their sensor-equipped mobile phones and builds indoor pathway maps without any a-priori knowledge of the building. We propose WiFi-Marks–defined using the tipping-point of an RSS trend–to overcome the challenges common to WiFi-based localization. Its location-invariant property helps to fuse user trajectories and make the system crowdsourcing-capable. We also present an efficient graph embedding algorithm that assigns optimal coordinates to the landmarks through a spring relaxation process based on displacement vectors. With the located WiFi-Marks and user trajectories, highly accurate pathway maps can be generated systematically. Our experiments demonstrate the effectiveness of Walkie-Markie.

12 Acknowledgement

We thank all the reviewers for their insightful comments and valuable suggestions, and Dr. Suman Banerjee for shepherding the final revision of the paper.

References

[1] Log-distance path loss model. http://en.wikipedia.org/wiki/Log-distance_path_loss_model.

[2] D. Amarasinghe, G. Mann, and R. G. Gosine. Landmark detection and localization for mobile robot applications: A multisensor approach. *Robotica*, 28:663–673.

[3] M. Azizyan, I. Constandache, and R. Roy Choudhury. Surroundsense: mobile phone localization via ambience fingerprinting. In *MobiCom '09*.

[4] P. Bahl and V. N. Padmanabhan. Radar: An in-building rf-based user location and tracking system. In *Infocom '00*.

[5] L. Bruno and P. Robertson. Wislam: Improving footslam with wifi. In *IPIN '11*.

[6] K. Chintalapudi, A. Padmanabha Iyer, and V. N. Padmanabhan. Indoor localization without the pain. In *MobiCom '10*.

[7] D.-K. Cho, M. Mun, U. Lee, W. J. Kaiser, and M. Gerla. Autogait: A mobile platform that accurately estimates the distance walked. In *PerCom 2010*, pages 116–124.

[8] I. Constandache, X. Bao, M. Azizyan, and R. R. Choudhury. Did you see bob?: human localization using mobile phones. In *MobiCom '10*.

[9] F. Dabek, R. Cox, F. Kaashoek, and R. Morris. Vivaldi: A decentralized network coordinate system. In *Sigcomm '04*.

[10] F. Evennou and F. Marx. Advanced integration of wifi and inertial navigation systems for indoor mobile positioning. *EURASIP J. Appl. Signal Process.*, pages 1–11, January 2006.

[11] B. Ferris, D. Fox, and N. Lawrence. Wifi-slam using gaussian process latent variable models. In *IJCAI'07*.

[12] Y. Gwon and R. Jain. Error characteristics and calibration-free techniques for wireless lan location estimation. In *MobiWac '04*.

[13] A. Haeberlen, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki. Practical robust localization over large-scale 802.11 wireless networks. In *MobiCom '04*.

[14] A. Hossain, Y. Jin, W. Soh, and H. Van. Ssd: A robust rf location fingerprint addressing mobile devices' heterogeneity. *Mobile Computing, IEEE Transactions on*, PP(99):1, 2011.

[15] A. Howard, M. Mataric, and G. Sukhatme. Relaxation on a mesh: a formalism for generalized localization. In *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems.*, volume 2, pages 1055 –1060, 2001.

[16] Y. Ji, S. Biaz, S. Pandey, and P. Agrawal. Ariadne: a dynamic indoor signal map construction and localization system. In *MobiSys '06*.

[17] A. R. Jimenez, F. Seco, C. Prieto, and J. Guevara. A comparison of Pedestrian Dead-Reckoning algorithms using a low-cost MEMS IMU. In *WISP '09*, pages 37–42.

[18] J. W. Kim, H. J. Jang, D. H. Hwang, and C. Park. A step, stride and heading determination for the pedestrian navigation system. *Journal of Global Positioning Systems*, 3(1-2).

[19] Y. Kim, H. Shin, and H. Cha. Smartphone-based wi-fi pedestrian-tracking system tolerating the rss variance problem. In *PerCom'12*.

[20] J. R. Kwapisz, G. M. Weiss, and S. A. Moore. Activity recognition using cell phone accelerometers. *SIGKDD Explor. Newsl.*, 12(2):74–82, Mar. 2011.

[21] F. Li, C. Zhao, F. Zhao, and et al. A reliable and accurate indoor localization method using phone inertial sensors. In *Ubicomp'12*.

[22] H. Lim, L. C. Kung, J. C. Hou, and H. Luo. Zero-configuration, robust indoor localization: Theory and experimentation. *Infocom '06*.

[23] D. Moore, J. Leonard, D. Rus, and S. Teller. Robust distributed network localization with noisy range measurements. In *SenSys '04*.

[24] J.-g. Park, B. Charrow, D. Curtis, J. Battat, E. Minkov, J. Hicks, S. Teller, and J. Ledlie. Growing an organic indoor location system. In *MobiSys '10*.

[25] N. B. Priyantha, H. Balakrishnan, E. Demaine, and S. Teller. Anchor-free distributed localization in sensor networks. Technical report, MIT CSail, 2003.

[26] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen. Zee: zero-effort crowdsourcing for indoor localization. In *Mobicom '12*.

[27] P. Robertson, M. Angermann, and M. Khider. Improving simultaneous localization and mapping for pedestrian navigation and automatic mapping of buildings by using online human-based feature labeling. In *IEEE/ION PLANS 2010*.

[28] P. Robertson, M. Angermann, and B. Krach. Simultaneous localization and mapping for pedestrians using only foot-mounted inertial sensors. In *Ubicomp '09*.

[29] J. Scarlett. Enhancing the performance of pedometers using a single accelerometer. In *Application Note, Analog Devices*, 2007.

[30] Y. Shang, W. Ruml, Y. Zhang, and M. P. J. Fromherz. Localization from mere connectivity. In *MobiHoc '03*.

[31] H. Shin, Y. Chon, and H. Cha. Unsupervised construction of indoor floor plan using smartphone. *IEEE Trans on Systems Man and Cybernetics. Part C-Applications and Reviews*, 2011.

[32] U. Steinhoff and B. Schiele. Dead reckoning from the pocket - an experimental study. In *PerCom '10*.

[33] S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. MIT Press, 2005.

[34] A. W. Tsui, Y. Hsiang Chuang, and H. Hua Chu. Unsupervised Learning for Solving RSS Hardware Variance Problem in WiFi Localization. *Mobile Networks and Applications*, 14:677–691, 2009.

[35] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury. No need to war-drive: unsupervised indoor localization. In *MobiSys '12*, pages 197–210, New York, NY, USA, 2012. ACM.

[36] O. Woodman and R. Harle. Pedestrian localisation for indoor environments. In *UbiComp '08*.

[37] K. Yamanaka, M. Kanbara, and N. Yokoya. Localization of walking or running user with wearable 3d position sensor. In *ICAT '07*.

[38] Z. Yang, C. Wu, and Y. Liu. Locating in fingerprint space: wireless indoor localization with little human intervention. In *Mobicom '12*.